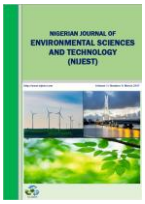
Nigerian Journal of Environmental Sciences and Technology (NIJEST)

www.nijest.com

ISSN (Print): 2734-259X | ISSN (electronic): 2734-2603

Vol 6, No. 1 March 2022, pp 13 - 27



Design of a Robust Orthogonal Frequency Division Multiplexing (OFDM) Transceiver for a Cognitive Radio Testbed

Bello N.[1,]* and Edeko F. O.[2]

[1,2]Department of Electrical/Electronic Engineering, University of Benin, Benin City, Edo State, Nigeria

Corresponding Author: *nosabello@uniben.edu

https://doi.org/10.36263/nijest.2022.01.0301

ABSTRACT

Primarily, a Cognitive Radio (CR) resolves the problem of spectrum scarcity and underutilization. However, it should also attain a good throughput and Quality of Service (QoS) as its objectives. These requirements are solely dependent on the modulation used. Though single carrier modulations such as Gaussian Minimum Shift Keying (GMSK) are highly spectral efficient, they are not robust to frequency selective fading. An implementation of multicarrier modulation supports high spectral efficiency and resistance to multipath fading. Therefore, in this paper, there is a comprehensive how-to-guide of the design of Orthogonal Frequency Division Multiplexing (OFDM) in cognitive radio using GNU Radio (GR) 3.8 that interfaces with Universal Software Radio Peripheral (USRP) B210s for video transmission. The implementation achieved allows for the choice of any modulation schemes out of four commonly used ones for communication of the payloads thus offering dynamism for use in a cognitive radio system. The effective implementation of the proposed methods in this paper is verified by the successful transmission and reception of a 4 minute 37 seconds MP4 video across the different choices of modulation schemes. It was observed that the error performance of the payloads degraded with higher constellation points however, the throughput was increased. Hence, the trade-off between error probability and throughput in the cognitive radio is based on the radio-scene metrics about the signal-to-noise ratio (SNR) of the available channels.

Keywords: OFDM, GNU Radio, SDR, USRP, VLC, wireless streaming, cognitive radio, GStreamer

1.0. Introduction

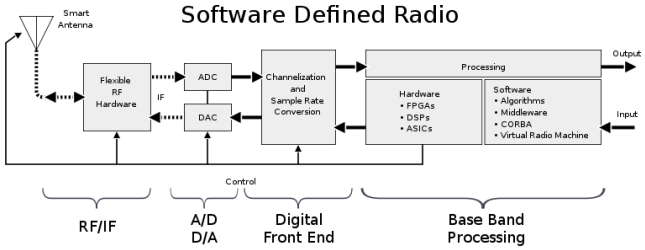
Haykin (2005) defined the cognitive radio as "an intelligent wireless communication system that is aware of its ambient environment. This cognitive radio will learn from the environment and adapt its internal states to statistical variations in the existing radio frequency (RF) stimuli by adjusting the transmission parameters (e.g., frequency band, modulation mode, and transmit power) in real-time and [in an] on-line manner". This intelligence avoids interference amongst users of the same wireless channel and provides the ability to use up all available bandwidth on the RF spectrum (TechTarget, 2008). To perform the adaptive function in a wireless environment, the architecture of the CR requires reconfigurable frontend parameters in software and less hardware. In this regard, the architecture of the CR follows a different protocol from the conventional open system interconnection (OSI) layer protocol. The Physical (PHY) layer or the RF frontend is the Software-Defined Radio (SDR) whereas the layers/protocols with adaptive protocols are the medium access control (MAC), Network, Transport protocols, then there is the cognitive radio control. Recent research in the development of Cognitive Radios (CRs) have used Gaussian Minimum Shift Keying (GMSK) (Singh, et al., 2016; Zhifeng Chen, 2011; Nimmi, et al., 2014) in the PHY layer which is known for its spectral efficiency but there is also the problem of multipath fading for which the robust Orthogonal Frequency Division Multiplexing (OFDM) scheme is designed to address. Researchers have explored the OFDM design with SDRs and discovered high error performance and data rates compared to single carrier modulations. Also, they have tested the design with different modulation. However, it has been

difficult to find a robust OFDM framework design for a cognitive radio system that uses the latest upgrades of commonly used software. Thus, in comparison, the single carrier simple GMSK framework has been used extensively in cognitive radio research with GNU Radio (GR) and the research on OFDM implementation has not been fully dealt with. Besides, there is an urgent need to train the next generation engineers and scientists in the cross-fields of multimedia and communications due to the growing demand for smart wireless systems and interest in multimedia social networks. In this paper, a successful setup of a testbed that supports developing a cognitive radio network for multimedia communications using the OFDM at the PHY was done.

2.0. Background

2.1. Software-defined Radio (SDR)

An SDR is made up of the RF front end and the data engine which is shown in Figure 1. The RF front end is the analogue part of the SDR while the data engine is the digital part.



ADC: analogue to digital converter ASIC: application-specific integrated circuit
DAC: digital to analogue converter DSP: digital signal processing
FPGA: Field Programmable Gate Array IF: intermediate frequency
RF: radio frequency

Figure 1: Software-defined radio (Wikipedia, 2017).

The data engine section (digital front end plus baseband processing) is usually implemented using microprocessors or Field Programmable Gate Arrays (FPGAs). It performs certain functions that can optionally perform digital signal processing (DSP) operations on digitized pre-filtered RF data before outputting the packed data over USB or Ethernet to the Host computer. It also handles sample rate mismatch using digital down-converters (DDC) and digital up-converters (DUC).

The RF section is the entire analogue section apart from the analogue to digital (ADC) and digital to analogue (DAC) converters. The interfaces of this section sit between the analogue outside world of RF and the rest of the digital radio. The section contains all front-end protection, filters, attenuators, transmitter filters, power amplification, and antenna switching. Common SDRs used in CR research are the RTL-SDR, HackRF One SDR, Ettus Research Universal Software Radio Peripheral (USRP), SDR#, etc.

2.2. Ettus Research USRP B210

Due to the popularity in cognitive radio research and compromise between cost and quality, the USRP B210 was used and therefore the specifications will be discussed briefly. The USRP B210 developed by Ettus Research (Research, 2021) is a low-cost SDR platform that utilizes a general-purpose processor and has gained widespread usage. USRP consists of two parts, a fixed motherboard and a plug-in daughterboard. The motherboard mainly contains ADC/DAC, an FPGA mainly for digital down sampling with programmable decimation rate and an interface connected to the host PC.

The daughterboard provides basic RF front-end functionality. The integrated RF frontend on the USRP B210 is designed with the new Analog Devices AD9361, a single-chip direct-conversion transceiver, capable of streaming up to 56 MHz of real-time RF bandwidth. The B210 uses both signal pipelines of the AD9361 thereby providing coherent 2×2 MIMO capability. Onboard signal processing and control of the AD9361 is performed by the Spartan6 XC6SLX150 FPGA connected to a host PC using USB 3.0. The USRP B210 real-time throughput is benchmarked at 61.44MS/s quadrature, providing the full 56 MHz of instantaneous RF bandwidth to the host PC for additional processing using GNU Radio or applications that use the USRP Hardware Driver (UHD) API (driver).

The board of the USRP B210 is shown in Figure 2. UHD is available for all major platforms including Linux, Windows, and can be built with many popular compilers such as GCC (Research, 2021; Liu, et al., 2011).

Figure 2: USRP B210 Board only (Research, 2021)

Several researches in cognitive radio reflect the use GNU Radio (GR) with USRP. GNU Radio (GR) is an open-source software providing various signal processing blocks accompanied with a graphical user interface. Other software platforms such as Simulink and LabView are also readily available (Liu, et al., 2011; Instrument, 2021).

2.3. GNU Radio (GR)

GNU Radio is a free and open-source software development toolkit that provides signal processing blocks to implement software-defined radios and signal-processing systems. It is a highly modular and flowgraph-oriented framework that comes with a comprehensive library of processing blocks easily combined to make complex signal processing applications. It enables users to design, simulate and deploy highly capable real-world radio systems and is widely used in hobbyist, academic and commercial environments to support both wireless communications research and real-world radio systems.

GNU radio is primarily supported on the Linux platform and when installed, comes with a large variety of tools and programs which can be used with USRP software-defined radios after installing the UHD.

GNU Radio Companion (GRC) is a graphical tool for creating signal processing flow graphs and generating flow-graph source code. Once the flowgraph is created, GRC creates a Python or C++ script from the graph. The script allows for further implementation and additions the user wants to add for more control over the signal flow. This script also provides the entire power and functionality of Python or C++ and its libraries, such as SciPy or NumPy for Python-centric processing of your signals or your favourite widget library to create any GUI you wish.

As described in the Guided Tutorials (Seeber, 2014). GNU Radio allows for the easy development of custom out of tree (OOT) blocks developed in C++ and wrapped in Python. The blocks can be used in any GNU Radio flowgraph. Many of these blocks are available on the Complementary GNU Radio Archive Network (CGRAN) website.

2.4. Orthogonal Frequency Division Multiplexing (OFDM) system model

Orthogonal Frequency Division Multiplexing (OFDM) is popular for high data rate applications which stem primarily from its efficient and flexible management of inter-symbol interference (ISI) in highly dispersive channels. It should be noted that the channel delay spread τ becomes an increasingly large multiple of the symbol time Ts—for example, due to significant multipath—the ISI becomes very severe. By definition, a high data rate system like LTE will generally have $\tau \gg Ts$ since the number of symbols sent per second is high. In a non-line of sight (NLOS) system like LTE that must

transmit over moderate to long distances, the delay spread will also frequently be large. This makes the use of a robust ISI-suppression technique mandatory. The theory of OFDM as a form of broadband multicarrier modulation method splits a high data rate signal stream into a multiple-smaller sub-signals that are transmitted simultaneously at different orthogonal frequency sub-carriers as shown in Figure 3.

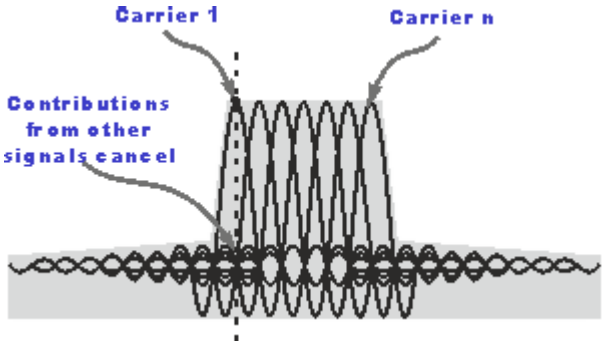


Figure 3: Orthogonal sub-carriers in the OFDM spectrum (CableFree, 2020)

This elegant development solves the problem of ISI provided the number of subcarriers is chosen to make the symbol time on each sub-stream much greater than the delay spread of the channel or, equivalently, to make the sub-stream bandwidth less than the channel coherence bandwidth. Let us suppose the use of L – orthogonal subcarriers in the frequency domain to transmit a block of serial data symbols in parallel (Rouphael, 2009). This is done by mapping (assigning) a carrier to each data symbol before transforming the data into the time domain via an Inverse Fast Fourier Transform (IFFT). The L-source symbol with period T_s each are buffered into a block of L-parallel modulated OFDM symbols with period:

$$T = LT_s. \tag{1}$$

Then the formed OFDM symbol vector, X is cyclic extended to have a length

$$L(1 + G), \tag{2}$$

where G is the fractional overhead (i.e., $v = GL$, v is the number of guard symbol).

The addition of a cyclic prefix or redundant symbols of length v to the transmitted OFDM signal appears to be a circular convolution of the transmitted signal with the channel. This creates an ISI free channel for the transmitted OFDM symbol.

At this point, the longer vector according to Equation (2) is then parallel-to-serial (P/S) converted into a wideband digital signal that can be amplitude modulated with a single radio at a centre frequency of

$$\omega = 2\pi f. \tag{3}$$

The transmitter OFDM model is shown in Figure 4 which captures all translation of the data symbols to OFDM symbols with cyclic prefix for multipath fading.

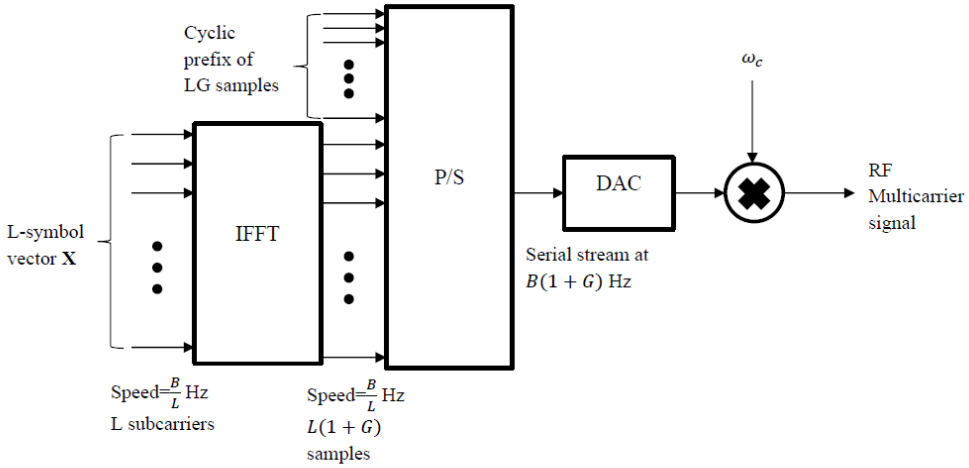


Figure 4: OFDM transmitter

At the receiver, the cyclic prefix is discarded, and the L received symbols are demodulated using an FFT operation, which results in L data symbols, each of the form

$$Y_l = H_l X_l + N_l \text{ for subcarrier } l. \quad (4)$$

Each subcarrier can then be equalized via a frequency equalizer by simply dividing by the complex channel gain $H[i]$ for that subcarrier. This results in Equation (5).

$$\widehat{X_l} = X_l + \frac{N_l}{H_l}. \quad (5)$$

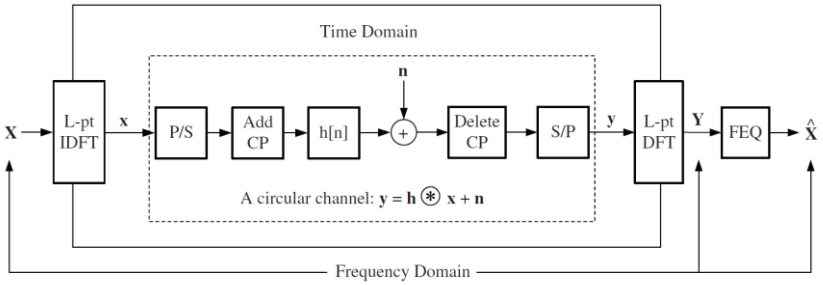


Figure 5: OFDM system configuration

In the above discussion, it was assumed that the transmitter and receiver are perfectly synchronized, and that the receiver perfectly knows the channel to perform the frequency equalization. In order to demodulate an OFDM signal, there are two important synchronization tasks that need to be performed by the receiver. They are the timing and frequency synchronization. The timing synchronization deals with correcting the timing offset of the symbol. Whereas, the frequency synchronization aligns the carrier frequency of the OFDM signal as closely as possible with the transmitted carrier frequency. Many synchronization algorithms have been developed in the literature and generally, the methods can be categorized as based on either pilot symbol or blind – cyclic prefix.

Pilot symbols are used in many OFDM communication systems and they are known symbols that are sent along with the transmission subcarriers. The pilot signals will be known by the receiver, and by looking for them in the received signal all parameters regarding synchronization and equalization will be deduced. The density of pilots in the signal will be proportional to the quality of the synchronization process, but it will also be inversely proportional to the amount of data transmitted,

therefore it creates a trade-off for the designer to consider. That is the reason why choosing a convenient density of pilots is an important part of the design of the application. Obviously, the quality and variance of the channel will be the key factors that will influence the amount of redundancy dedicated to synchronization in the form of pilots.

The blind approach does not include the pilot symbols, so in the second category, the receiver must do the best it can without explicitly being able to determine the effect of the channel. In the absence of pilot symbols, the Cyclic Prefix (CP), which contains redundancy, can also be used to attain time and frequency synchronization (Schmidl & Cox, 1997). This technique is effective with a large number of subcarriers or when the estimation of the offset is done over several consecutive symbols. The principal benefit of CP-based methods is that pilot symbols are not needed, so the data rate can nominally be increased. In Wi-MAX, accurate synchronization, and especially channel estimation, are considered important enough to warrant the use of pilot symbols, so the blind techniques are not usually used for synchronization.

Samijayani *et al*. (2017) designed a video transmission using SDRs and GR 3.7. The video transmission implementing OFDM was tested with the parameters of the FFT length of 128, 256, and 512 as well as with the different types of modulation. According to Samijayani *et al*. (2017), the video transmission with the best result was performed using 512 FFT lengths. Nguyen (2013) uses an SDR testbed to evaluate the practical error performance of OFDM based systems in different propagation conditions. The testbed is built based on the GR software platform and USRP devices. The theoretical results in simulation show that OFDM provides good error performance and high data rate transmission compared to single carrier modulations techniques. But, in practice, the OFDM testbed was unexplored, so, the evaluation in simulations was made in terms of the technical demonstration of OFDM implementation, plots of error performance curves with different modulation schemes, in-depth analyses of synchronization and channel estimation processes and their effects to experimental performances, and analyses of realistic limitations of SDR and USRP. These researches provide the basics for the OFDM implementation through the highlights of the expected results with SDR development. They suggest the use of several modulation schemes on the OFDM signal and powerful computers to handle processing to reduce processing overhead and errors. However, they do not provide the means to develop a foundational application framework with OFDM that supports a substantial starting point for CR research just like that of GMSK which have been extensively dealt with (Ghani, et al., 2017). Ghani et al, (2017) achieved a video transmission framework that implemented OFDM. The OFDM supported four modulation schemes which are the BPSK, QPSK, 16QAM, and 64QAM at code rates values of ½, 2/3, and ¾. The research used GR and OpenCV software, USRP B200 SDR, and two host computers each having a Core i7 2nd generation processor with 4GB RAM. This paper aims to design a similar OFDM framework with GR 3.8, GStreamer, and VLC player for a 4 minutes and 37 seconds video transmission and also offer ways to make modifications to the framework towards the implementation of a CR protocol.

3.0. Methodology

The research on OFDM implementation presented in this paper was carried out using a core i7 8th generation laptop with 16 GB RAM and a Core i5 laptop with 8GB RAM for the receiver and the transmitter ends respectively. Both laptops had Ubuntu 18.04 LTS and one had an additional operating system, Parrot 4.7 security. All operating systems had GR 3.8 using USRP hardware driver (UHD) v3.15.0. To be able to transmit and receive a file's contents between two USRPs, the GNU Radio flowgraph was constructed. The structure of the OFDM system in GR 3.8 is similar structure as that found in the research (Samijayani, et al., 2017; Nguyen, 2013) with a nuanced version of the PHY layer implementation. First, the section will deal with the guide towards a successful GR flowgraph for the OFDM transmitter and receiver.

3.1. Data transmission flow

Data transmission starts with a File Source block which reads raw data bytes from a specified file. After the File Source block is the Stream to Tagged Stream block for fragmentation. Each fragment of data is tagged with its size using a tag key of string type. After the tagged stream of a packet is generated, the Cyclic Redundancy Check (CRC) is appended before the packet. Next in the pipeline is

the protocol formatter block used for generating the header. It takes a formatter object of the digital.packet_header_ofdm class. Once the header bytes have been generated, the bytes are then mapped to the constellation space using any modulation scheme of interest. BPSK was used for the header symbol mapping due to its resilience to noise. It is done in two steps; repacking the bytes to bits depending on the number of constellation points and mapping the repacked bits to the respective constellation points. So, the header bytes are repacked 8 bits to 1 bit or 1 byte to 1 bit and then the chunk to symbols block maps the bits to the respective constellation points using gray coding. The same is repeated for the payload data using the default digital modulation technique, QPSK. Next in the sequence is the Tagged Stream Mux block which multiplex the header and payload in this order to form a complete packet. The block achieves this using the tag key created earlier.

At this point, the data symbols have been generated, so we perform OFDM on these packets or payload symbols. First, in the generation of the OFDM signal, the OFDM frame has to be generated consisting of a repeated preamble and the payload symbols on the available subcarriers. This is done with the OFDM Carrier Allocator block, just before the Inverse Fast Fourier Transform (IFFT) is carried out on the OFDM symbol. The block handles the distribution of complex symbols in the frequency domain and adds pilot symbols and preambles. After the OFDM symbol is generated, the Cyclic Prefix (CP) is added to correct channel multipath interference. The OFDM Cyclic Prefixer block handles the task of adding the CP and the roll-off in the OFDM signal. The Multiply Const block was used to constrain the dynamic range within ± 1 to avoid saturation of the power amplifiers and therefore avoid the nonlinearity problem in the DDC of the USRP B210. The transmitter and receiver flowgraphs are shown in Figures 6 and 7.

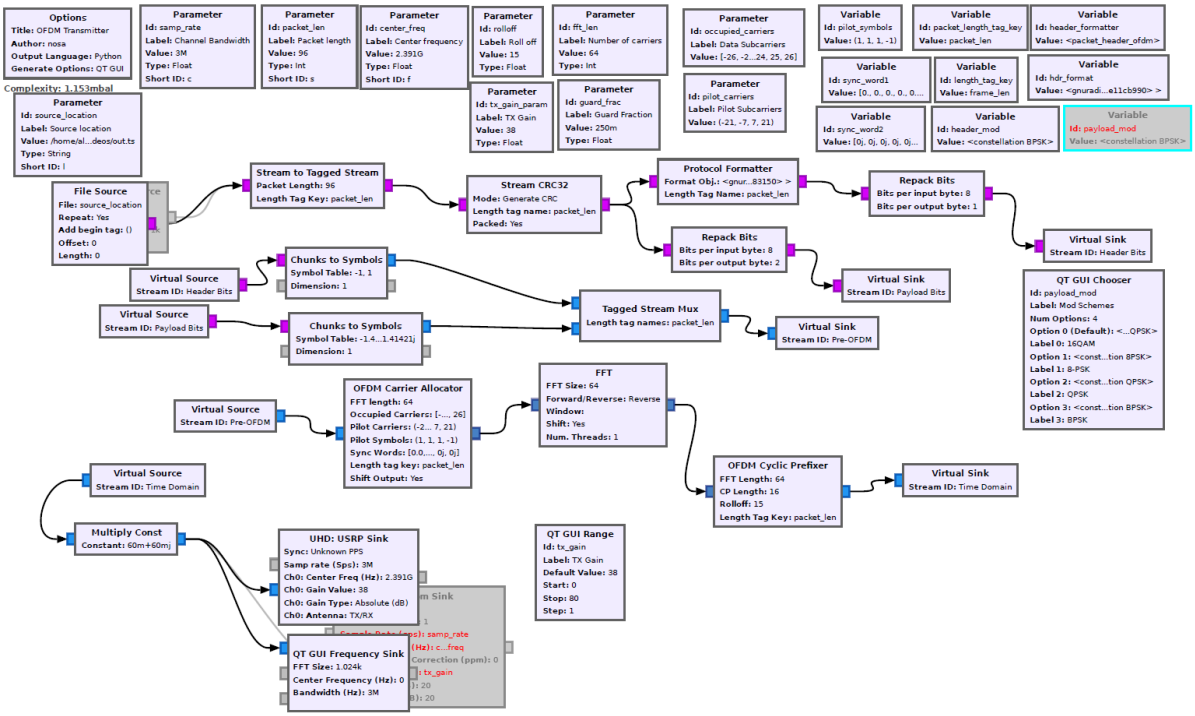


Figure 6: OFDM transmitter flowgraph

3.2. Data receiver flow

On the OFDM receiver end, there are three steps to perform, the detection of the start of an OFDM frame amidst the timing offset of the received OFDM signal, frequency and timing synchronization or correction. The Schmidl and Cox OFDM Synch. block, together with the frequency Mod block, Delay and Multiply block are used for frame synchronization. In the Schmidl and Cox Synch. block, the Frequency offset is calculated which is later used by the Serializer block for frequency offset correction. The trigger signal is used by Header/Payload Demux block for demultiplexing the header and the payload of the OFDM frame through the trigger or trigger tag it receives. Once the trigger is received, the header is sent to be decoded to obtain the payload information. The pipeline for

decoding the OFDM header is first through the FFT block and respectively through the OFDM Channel Estimation, Equalizer and Serializer blocks. The Channel Estimation (Anjana, et al., 2015) block estimates the channel used and the coarse frequency offset from the preambles. Next, the Frame Equalizer block performs the equalization in one or two dimensions of the OFDM frame. Lastly, for OFDM demodulation, the Serializer plucks the preamble symbols and outputs streams of the complex header symbols. With the header symbols in place, the bits are generated using the constellation decoder and next to it is the packet header parser for decoding the hard bits of the header. The parsed header contains information such as the length of the payload which is passed to the Header/Payload Demux block to produce the payload. The payload pipeline takes the same decoding process which is similar to that of the header except it does not contain the Channel Estimation block and repacks the bits to bytes when its symbols have been decoded then checks for CRC before saving packets to file.

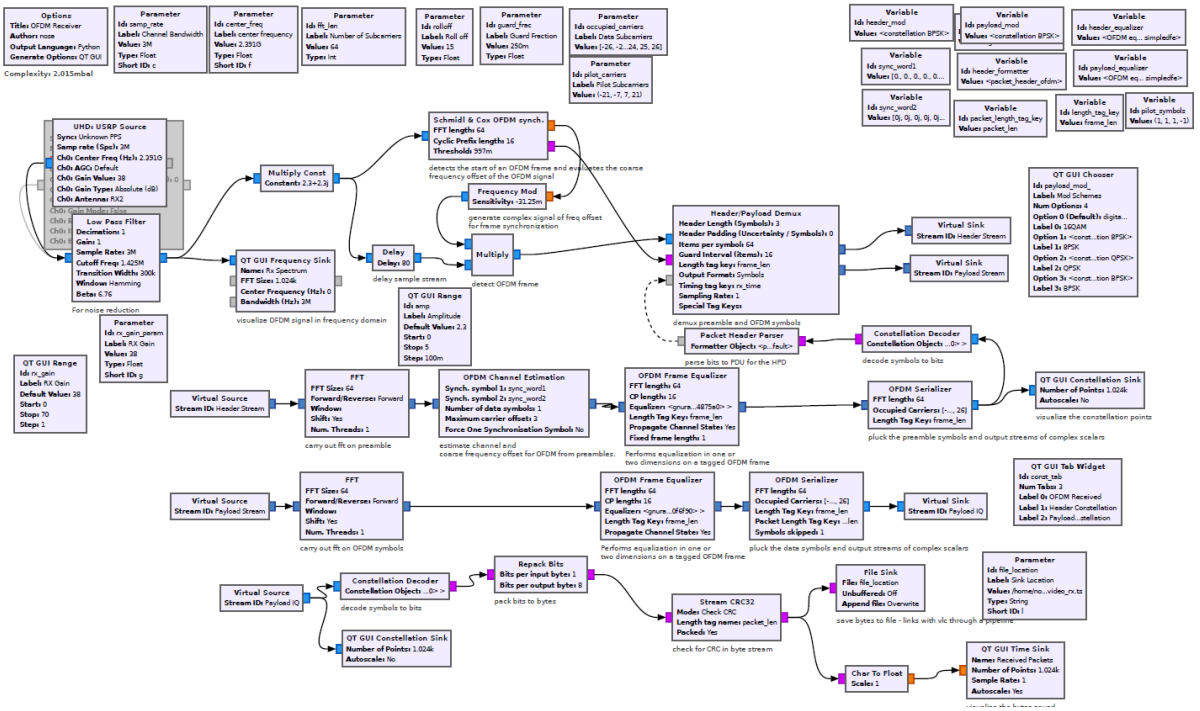


Figure 7: OFDM receiver flowgraph

3.3. Changing parameters from command line

For the file transfer blocks, a few parameter blocks were defined. A parameter block represents a parameter to the flowgraph. A parameter can be used to pass command line arguments into the OFDM programs with the python files. The parameters within reach from the terminal are the file source, sample rate (or bandwidth), packet size, number of carriers (or FFT length), roll off, guard fraction, centre frequency, pilot carriers, transmitter gain, and receiver gain as shown in Figures 8 and 9. The choice of modulation scheme can be changed during runtime from the choice provided in the interface.

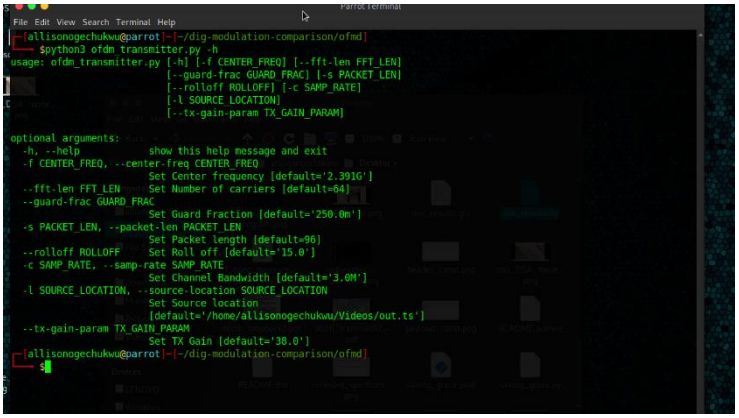


Figure 8: Changing the parameters from the command line for the OFDM transmitter

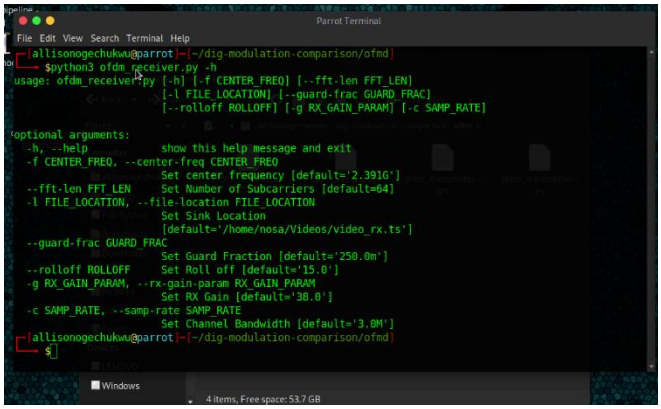


Figure 9: Changing the parameters from the command line for the OFDM receiver

3.4. OFDM interface

The interface showing the loop back of the OFDM design is given in Figure 10. It contains received signals, the constellation diagrams of the transmitted header and payloads, the options for changing modulation schemes in real-time, the received video packets and video playback in the small VLC window by the side. The parameters used for the demonstration is given in Table 1.

Table 1: Values used in OFDM design

Symbol	Description	Relation	Design values
B	Nominal Bandwidth	Sampling rate	3 MHz
B_{chan}	System channel bandwidth	Channel spacing	
L	Number of carriers	Size of FFT and IFFT	64
–	Occupied Carriers	-	52
L_d	Data subcarriers	L – pilot subcarriers	48
G	Guard Fraction	% of L	0.25
–	Modulation techniques	-	BPSK, QPSK, 8-PSK, and 16QAM
Δf	Subcarrier spacing	Independent of L	
N_g	Guard symbols	$N_g = GL$	16
T	OFDM symbol time	$T = \frac{L + N_g}{B}$	$26.67\mu s$

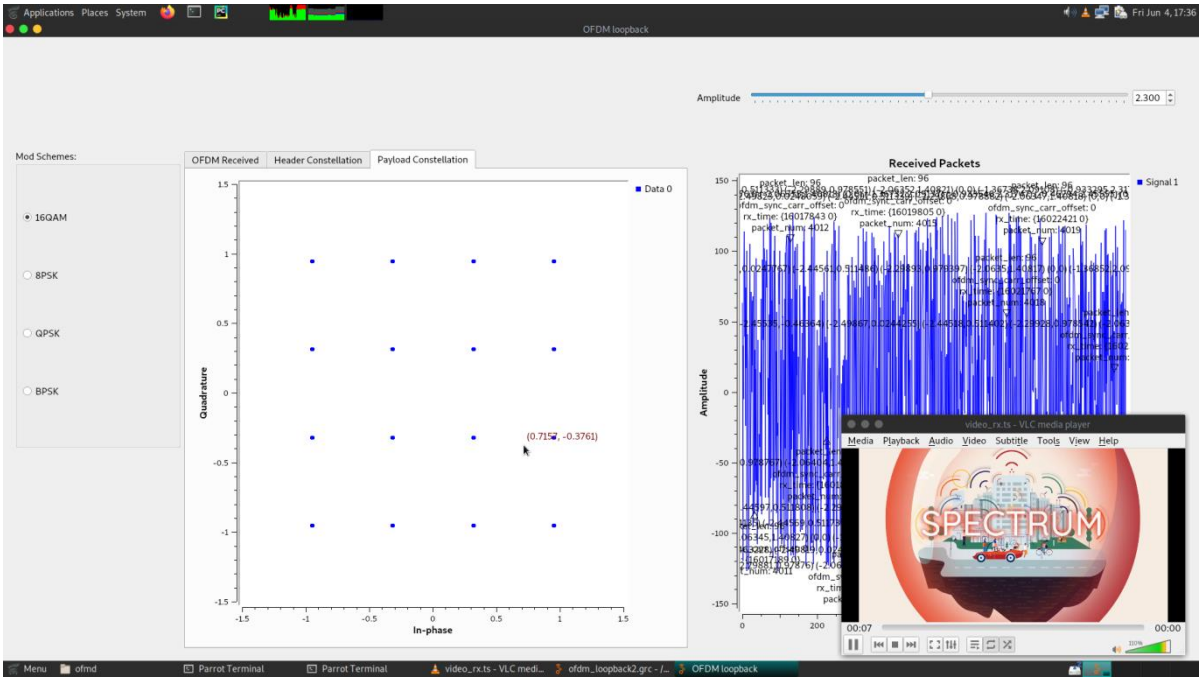


Figure 10: GNU Radio (GR) 3.8 OFDM interface

Our final hardware configuration for this project utilized two Ettus Research USRP B210s and the VERT900 antennas. Our final implementation can be seen in Figures 11 and 12.

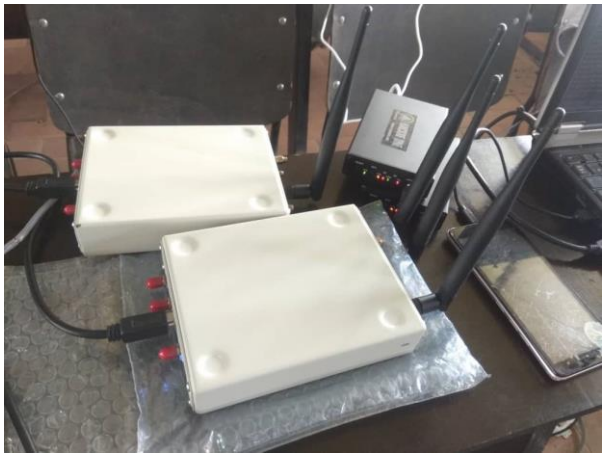


Figure 11: Setup for OFDM transmission

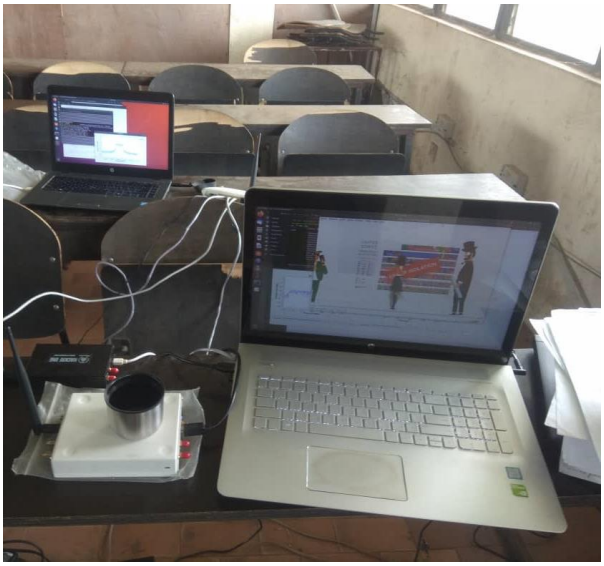


Figure 12: Setup for OFDM transmission

3.5. OFDM setup procedure

The visual result is the output of the GR, GStreamer, and VLC. The OFDM design is one part of the entire setup for a working video transmission. Usually, the video data would be in MP4 format which is not compatible for video broadcast or transmission. The transport stream (TS) format provides chunks of data with error correction and detection methods so that in the case of data corruption or loss the receiver will recover from the corrupt frames perhaps by dropping some frames. Gstreamer is used for transcoding and in the research, it was used for transcoding the MP4 to TS format. The command used for video and audio transcoding is given in Figure 13.

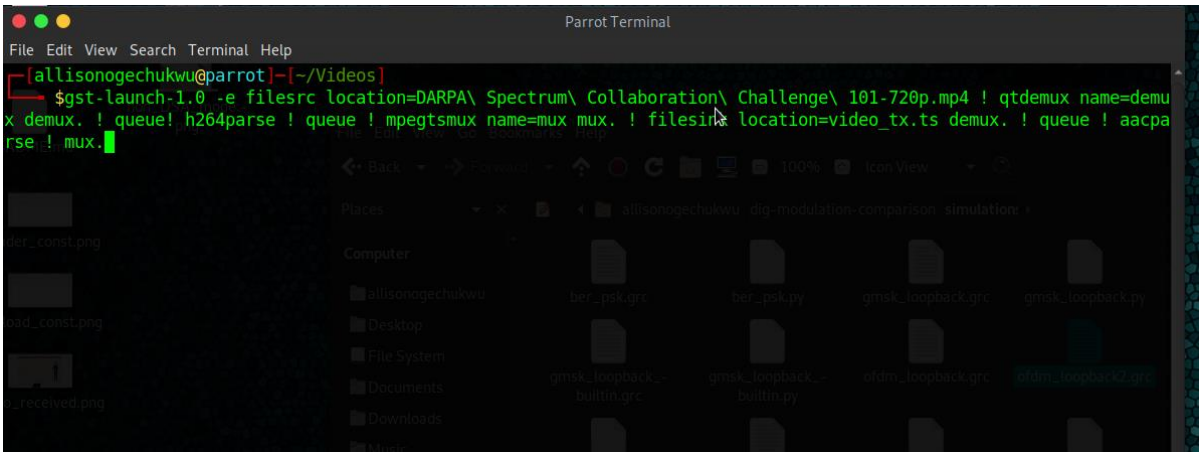


Figure 13: GStreamer command for transcoding from MP4 to TS format

VLC player, on the other hand, plays the received packets in TS format. Gstreamer and mplayer could have been used for transcoding instead (Nimmi, et al., 2014; Zhifeng Chen, 2011; Singh, et al., 2016), but VLC showed better performance with serious corrupted data. A Linux pipeline was used to link the two ends together, the GR program for the OFDM receiver and the VLC player. This was done with the mkfifo command for a first-in-first-out (FIFO) pipeline.

To setup the OFDM transmission, the receiver should run first in anticipation of the transmitted data. Otherwise, it is difficult for the VLC player to correctly play the received packets and would take a while to recover. Preferably, the video should be on repeat to ensure a long transmission, which can be done from the File Source block in GR. Then, the transmitter should run afterwards for clear and satisfying video transmission. There are some parameters that can be adjusted to ensure successful transmission and they are, the gain of the transmitter and the receiver. It is advisable to start with small values and then increase the values based on the level of SNR desired at the receiver for correct reception of the OFDM signal. In summary, the steps are:

i. Transcode the MP4 format to TS format using GStreamer
ii. Create a FIFO pipeline with mkfifo <filename>
iii. Play the received file (FIFO buffer) with VLC player
iv. Start the OFDM receiver flowgraph
v. Start the OFDM transmitter flowgraph

In the above steps, it is assumed that the USRP devices are already connected to the host computers used for the test.

4.0. Results and Discussion

Shown in Figures 14 and 15 are RF spectrum of the OFDM signals. There is a difference in the SNR between both plots as the SNR in the receiver spectrum is lower due to noise from the RF environment. Also, to show the end-to-end delay (EED) in the communication system, we ran a transceiver file of the OFDM containing the transmitter and receiver flowgraph in one flowgraph. The result of the lag produced between the transmitted and received packets is shown in Figure 16. It shows a delay of about 80 sample points between the transmitted and received packets which in terms of time would depend on the sample rate used.

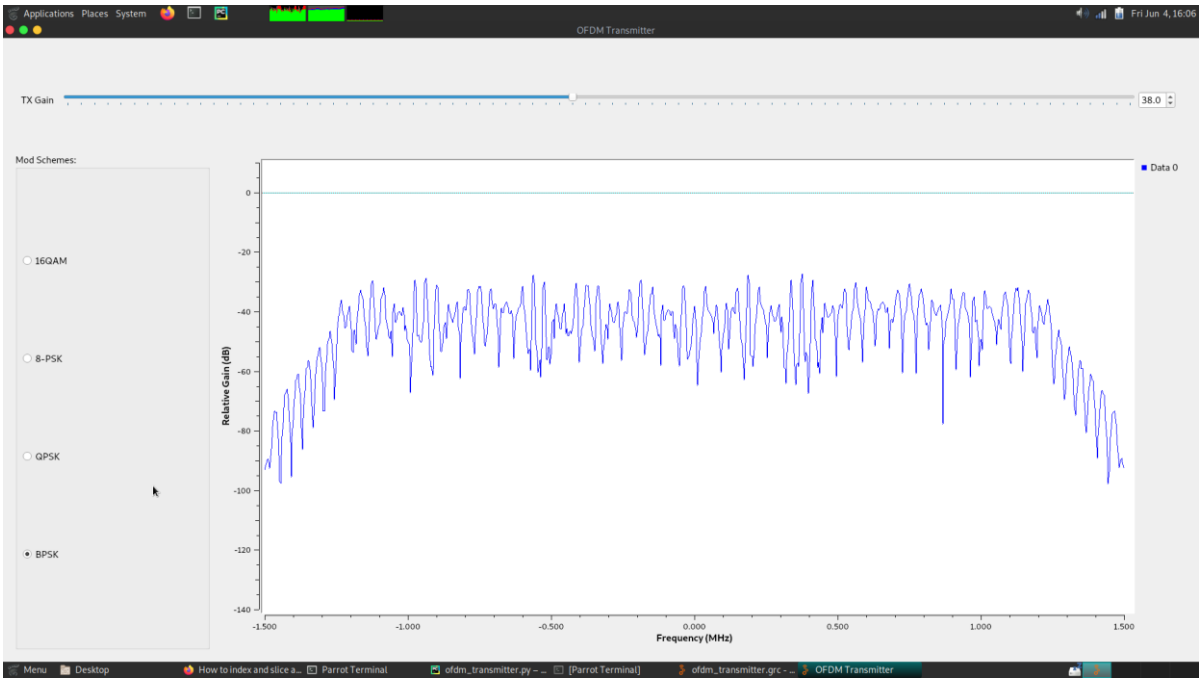


Figure 14: Transmitter OFDM signal

Since, higher samples rates are usually used in the degree of MHz, the EED would be in microseconds. The performance of the designed system is assessed by determining the data error rate during transmit and receive process. This was done through analysis of the saved file during testing. Table 2 presents the theoretical data rates and the measured and the attainable throughput of the system at different modulation schemes. The throughput puts into consideration the size of data saved, the time taken to play the received packets through VLC and the time taken to complete the transmission of a 4 minutes 37 seconds video.

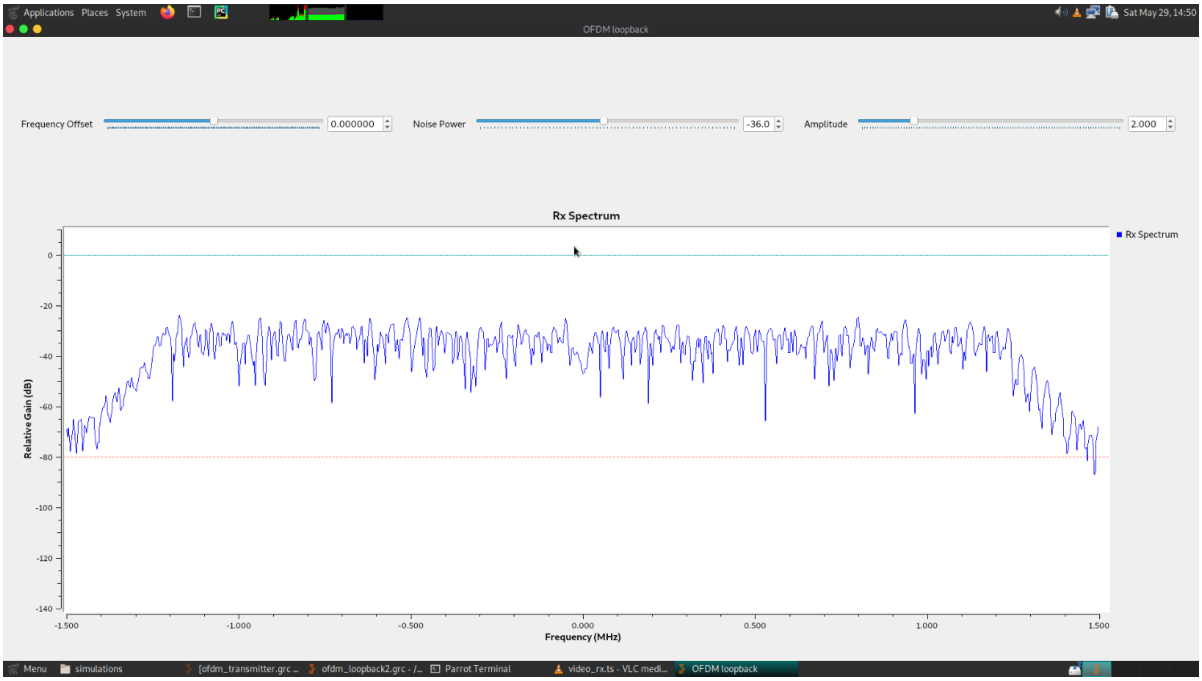


Figure 15: Receiver OFDM signal

Table 2: Theoretical data rates versus the throughput of the OFDM system at different modulation schemes

Modulation Schemes	BPSK	QPSK	8-PSK	16QAM
Theoretical data rate (Mbps)	1.8	3.6	5.4	7.2
Throughput (Mbps)	12.2	23.4	37.7	50.3

The result in Table 2 shows that the higher rate modulation schemes were more error-prone in a highly noisy channel or the presence of external interference occurring in the channel. Nevertheless, there is a correlation between the theoretically determined data rates for each modulation and the throughput achieved.

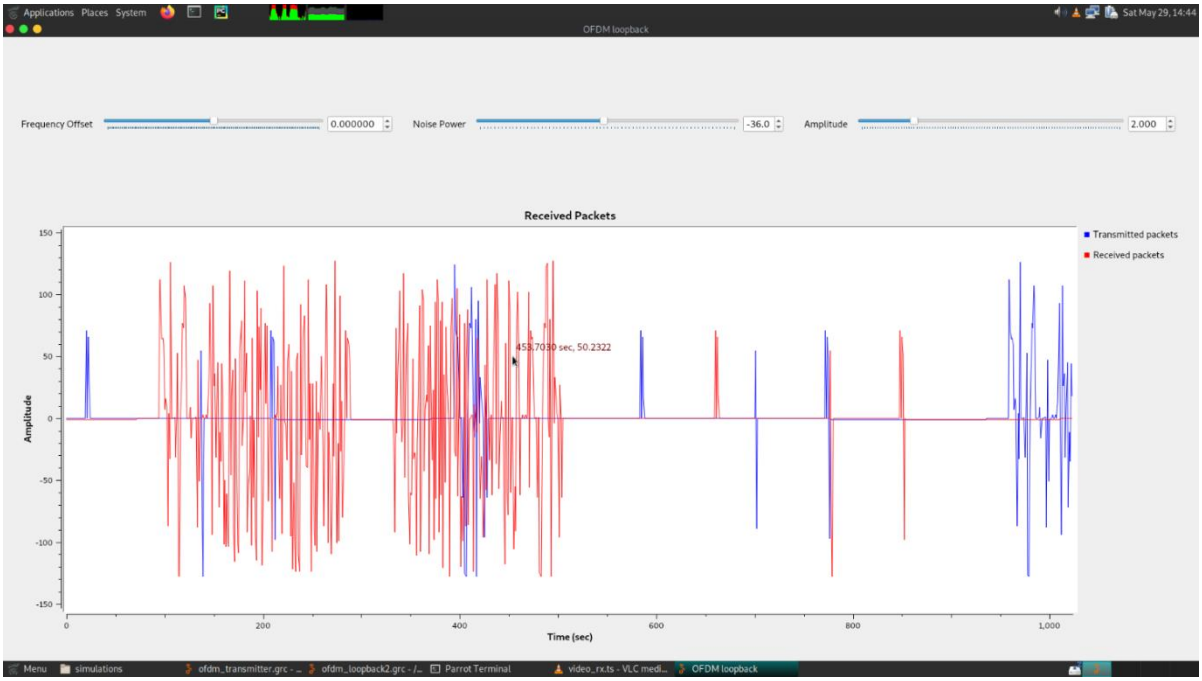


Figure 16: Transmitted and received bytes plot show that there is a delay in the received packets

4.1. Implementing a cognitive radio (CR)

The implementation of a cognitive radio system can be seen as a two-step process; the design of the PHY layer modulations (waveforms) and the data analysis part. The possibility of the CR to choose modulations provides optimum use of good Primary User (PU) channels. This testbed could be used to communicate the OFDM waveform signal of the CR with adaptable modulation schemes. In a CR system, the waveform choice is based on the information from the meters – the results of the radio scene analysis – gotten from the decision makers and optimizers (Rondeau, 2007). A cognitive radio model typically will contain several waveforms (PHY layer modulations such as the GSMK, OFDM, DBPSK, etc.) which would be selected to transmit on a channel of a certain bandwidth, channel capacity, interference temperature limit, etc. The testbed used in this research can be extended towards the CR research easily by making some modifications that implement the second part – the other modules of the CR. These modules are the spectrum sensing module, the decision-makers, the optimizers, the policy domain, the user interface, and the cognitive controller (Rondeau, 2007). All other modules besides the controller have specific objectives and functions, however, the cognitive controller (wrapper module) functions as the system kernel and scheduler that handles the input/output timing of the other modules. With the python generated file of the OFDM modulation layer, CR researchers can make the additional implementation of interfacing the SDR framework (all the PHY layer modulation waveforms) to the cognitive controller thereby implementing a CR node. Typically, a CR node as shown in Figure 17, in accomplishing its functions and opportunistically accessing unused channels, carries out a sequence of steps that are repeated in a cycle. This cycle is performed with the modules that make up the CR that has been discussed. The following are the steps taken by a CR node in each cycle that shows the information flow from the inputted meters (information) to the outputted knobs (actions) (Rondeau, 2007; Rondeau & Bostian, 2009; Le & Bostian, 2007; Le, et al., 2007; Scaperoth, et al., 2006):

i) The radio hardware collects information from the radio scene to provide the cognitive system with the environmental context, while other observations occur to read information about the user's needs, policy demands and hardware capabilities.
ii) All information is then processed into an understandable model of the current state and passed to the learning core, which comprises the decision-makers and the optimizers.
iii) The system is aided with the knowledge base to keep track of progress and behaviour overtime to help the optimization process find better solutions faster.
iv) The radio adapts to the new configuration and the cycle repeats to ensure the adaptation was appropriate and that no other environmental changes require further adaptation.

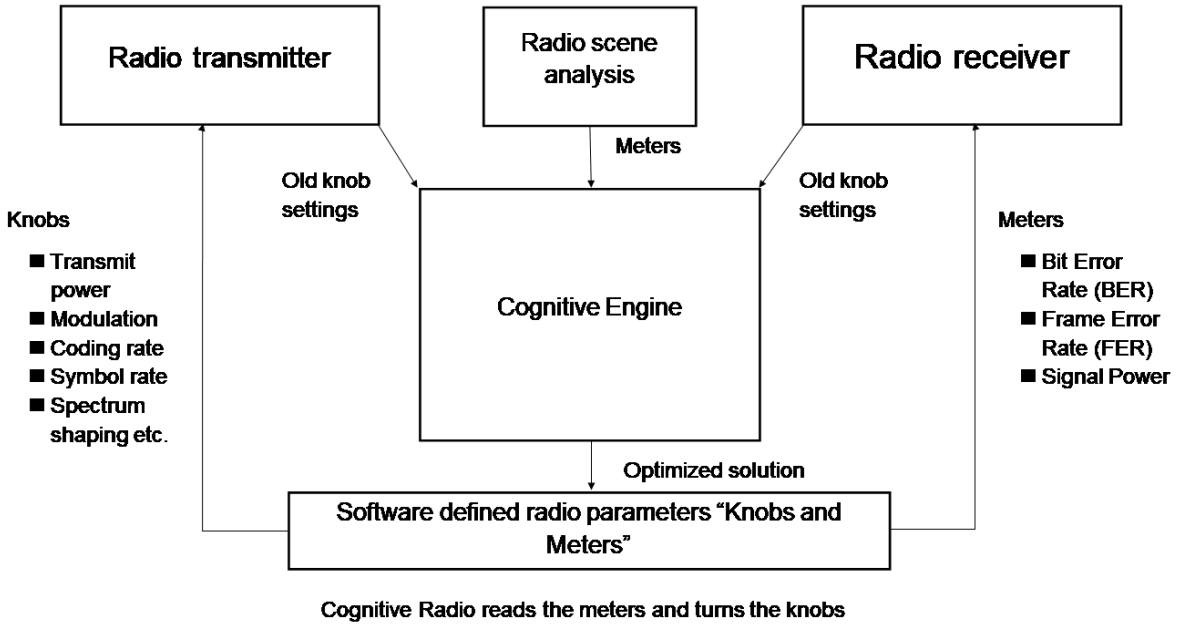


Figure 17: Block diagram of a Cognitive Radio (CR) node

The design structure that accomplishes the main functions and combines all the components is a designer's call which should optimally meets the ultimate goal.

5.0. Conclusions

The study has successfully continuously transmitted and received the contents of video file through an OFDM system that was designed with GR 3.8. The OFDM system supports real-time changes to the modulation scheme used on the payloads and buffer replay through VLC player. The higher modulation schemes offer high data rates at the cost of low probability of detection or noise resilience and the lower modulation schemes offer high probability of detection at the cost of lower data rates. The buffer system was useful for improved QoS as the video still played even when an interference had occurred in the channel. Also, the successful video transmission showed high quality video signal and better interference resilience when external interference affected the channel. The implementation can be easily modified to a CR system besides the video lag and quality degradation when higher modulation schemes are used because there is the possibility of smart choice of modulation by the CR depending on the capacity of the channel used when intelligently accessing the RF spectrum. We believe the proposed testbed will potentially benefit the research community for the study of various CR networks.

References

Anjana, C. et al., (2015). An experimental study on channel estimation and synchronization to reduce error rate in OFDM using GNU Radio. *Procedia Computer Science,* 46, pp. 1056–1063.

CableFree (2020). *OFDM (Orthogonal Frequency Division Multiplexing).* [Online] Available at: https://www.cablefree.net/wirelesstechnology/ofdm-introduction/ [Accessed 20 06 2021].

Ghani, M. F. A. et al., (2017). *Video transmission based on Orthogonal Frequency Division Multiplexing (OFDM) utilizing television white space.* s.l., s.n., pp. 152–157.

Haykin, S. (2005). Cognitive radio: brain-empowered wireless communications. *IEEE journal on selected areas in communications,* 23, pp. 201–220.

Instrument, N. (2021). *Select your LabView Edition.* [Online].

Le, B., Rondeau, T. W. and Bostian, C. W. (2007). Cognitive radio realities. *Wireless Communications and Mobile Computing,* 7, p. 1037–1048.

Le, T. W. R. B. and Bostian, C. W. (2007). *General radio interface between cognitive algorithms and reconfigurable radio platforms.* s.l., s.n.

Liu, W. et al., (2011). *Real-time wide-band spectrum sensing for cognitive radio.* s.l., s.n., pp. 1–6.

Nguyen, D. T. (2013). *Implementation of OFDM systems using GNU Radio and USRP,* s.l.: s.n.

Nimmi, S., Saranya, V., Gandhiraj, R. et al., (2014). *Real-time video streaming using GStreamer in GNU Radio platform.* s.l., s.n., pp. 1–6.

Research, E. (2021). *USRP B210 (Board Only).* [Online].

Rondeau, T. W. (2007). *Application of artificial intelligence to wireless communications,* s.l.: s.n.

Rondeau, T. W. and Bostian, C. W. (2009). *Artificial intelligence in wireless communications.* s.l.:Artech House.

Samijayani, O. N. et al., (2017). *Implementation of SDR for video transmission using GNU radio and USRP B200.* s.l., s.n., pp. 1–4.

Scaperoth, D. et al., (2006). *Cognitive radio platform development for interoperability.* s.l., s.n., pp. 1–6.

Schmidl, T. M. and Cox, D. C. (1997). Robust frequency and timing synchronization for OFDM. *IEEE transactions on communications,* 45, pp. 1613–1621.

Seeber, B. (2014). *GNU Radio Tutorials.* [Online]
Available at: https://wiki.gnuradio.org/index.php/Tutorials

Singh, A., Rani, S. and Kakkar, S. (2016). *Video Transmission through GMSK using GNU Radio.* s.l., s.n.

TechTarget (2008). *Cognitive Radio Definition.* [Online]
Available at: https://searchnetworking.techtarget.com/definition/cognitive-radio

Wikipedia (2017). *Software-defined radio.* [Online]
Available at: https://en.wikipedia.org/wiki/Software-defined_radio

Zhifeng Chen, J. X. (2011). *Cross-layer wireless video testbed.* [Online]
Available at:
http://www.wu.ece.ufl.edu/projects/wirelessVideo/project/H264_USRP/index.htm.

Cite this article as:
Bello N. and Edeko F. O., 2022. Design of a Robust Orthogonal Frequency Division Multiplexing (OFDM) Transceiver for a Cognitive Radio Testbed. *Nigerian Journal of Environmental Sciences and Technology*, 6(1), pp. 13-27. https://doi.org/10.36263/nijest.2022.01.0301